Python for Data Science

Correlation

Page sections:

Introduction

Correlation is a measure of relationship between variables that is measured on a -1 to 1 scale. The closer the correlation value is to -1 or 1 the stronger the relationship, the closer to 0, the weaker the relationship. It measures how change in one variable is associated with change in another variable.

There are a few common types of tests to measure correlation, these are: Pearson, Spearman rank, and Kendall Tau. Each have their own assumptions about the data that needs to be meet in order for the test to be able to accurately measure the level of correlation. These are discussed further in the post. Each type of correlation test is testing the following hypothesis.

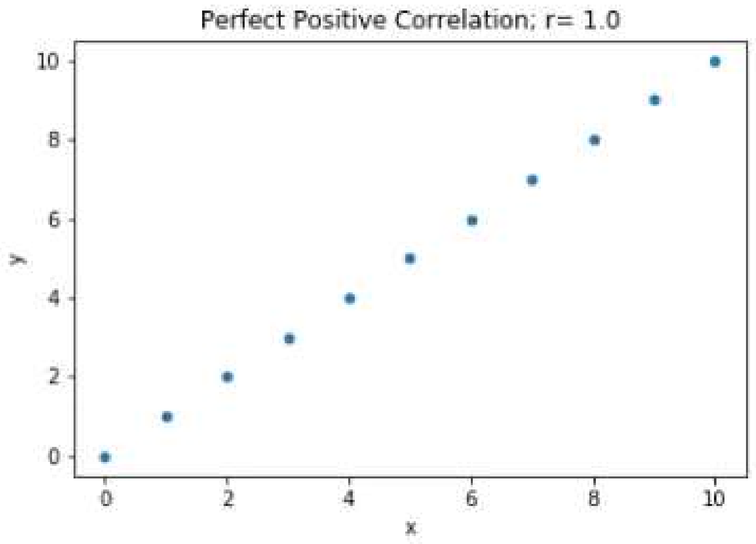
H_0 hypothesis: There is not a relationship between variable 1 and variable 2

H_A hypothesis: There is a relationship between variable 1 and variable 2

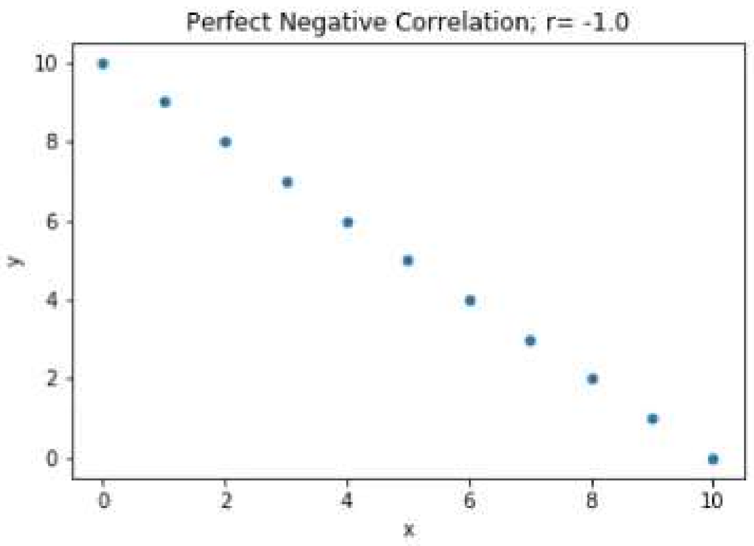
If the obtained p-value is less than what it is being tested at, then one can state that there is a significant relationship between the variables. Most fields use an alpha level of 0.05 which I will also use.

There are a few types of correlation:

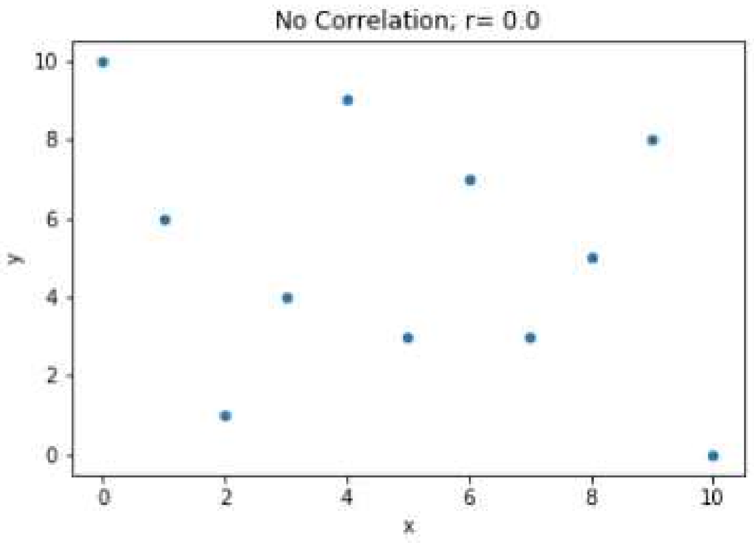
- **Positive correlation**: as one variable increases so does the other



- **Negative (inverse) correlation**: as one variable increases the other variable decreases



- **No correlation**: there is no association between the changes in the two variables



The strength of the correlation matters. The closer the absolute value is to -1 or 1, the stronger the correlation.

r value	Strength
0.0 – 0.2	Weak correlation
0.3 – 0.6	Moderate correlation
0.7 – 1.0	Strong correlation

Pearson correlation assumptions

Pearson correlation test is a parametric test that makes assumption about the data. In order for the results of a Pearson correlation test to be valid, the data must meet these assumptions:

- The sample is independently and randomly drawn
- A linear relationship between the two variables is present
  - When plotted, the lines form a line and is not curved

- There is homogeneity of variance

The variables being used in the correlation test should be continuous and measured either on a ratio or interval sale, each variable must have equal number of non-missing observations, and there should be no outliers present.

Spearman Rank correlation assumptions

The Spearman rank correlation is a non-parametric test that does not make any assumptions about the distribution of the data. The assumption for the Spearman rank correlation test is:

- There is a monotonic relationship between the variables being tested
  - A monotonic relationship exists when one variable increases so does the other

For the Spearman rank correlation, the data can be used on ranked data, if the data is not normally distributed, and even if the there is not homogeneity of variance.

Kendall’s Tau correlation assumptions

The Kendall’s Tau correlation is a non-parametric test that does not make any assumptions about the distribution of the data. The only assumption is:

- There should be a monotonic relationship between the variables being tested

The data should be measured on either an ordinal, ratio, or interval scale.

Data used for this Example

The data used in this example is from Kaggle.com which was uploaded by the user shivamagrawal. Link to the Kaggle source of the data set is here, or you can load it into pandas from our GitHub using the code shown a bit later.

For this example, I will test if there is a significant relationship between the carat and price of diamonds. These are variables "carat" and "price", respectively, in the data set. Now I will load the libraries, the data set, and take an brief look at the descriptive statistics for these variables.

```
# Importing required libraries
import pandas as pd
import researchpy as rp
from scipy import stats

df =
pd.read_csv("https://raw.githubusercontent.com/Opensourcefordatascience/Dat
a-sets/master/diamonds.csv")

df[["carat", "price", "depth"]].describe()
```

	carat	price	depth
count	53,940	53,940	53,940
mean	0.797940	3932.799722	61.749405
std	0.474011	3989.439738	1.432621

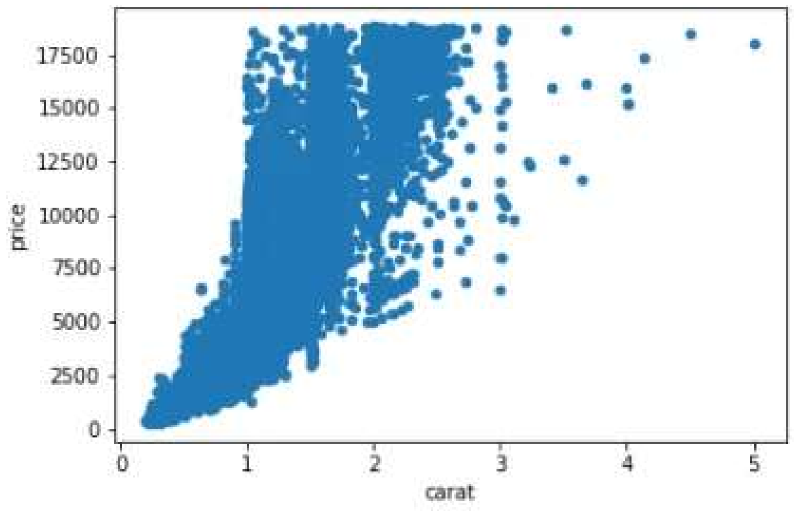
min	0.200000	326.000000	43.000000
25%	0.400000	950.000000	61.000000
50%	0.700000	2401.000000	61.800000
75%	1.040000	5324.250000	62.500000
max	5.010000	18823.000000	79.000000

Checking the Assumptions

Linear relationship

I will use the built-in method from pandas to plot a scatter plot to look for a linear relationship.

```
df.plot.scatter("carat", "price")
```



It appears that there is a linear relationship present- as the carat increases so does the price. It’s a bit hard to tell because there is a shotgun shot looking spread that starts after 1 carat. This leads me to believe that we are violating the assumption of homoscedasticity between the variables.

Homogeneity of variances

To formally test homogeneity of variances, I will use the Levene's test of homogeneity of variances which is the stats.levene() method from scipy.stats. Full documentation on the method can be found here.

```
stats.levene(df['carat'], df['price'])
```

```
LeveneResult(statistic=40965.266273805122, pvalue=0.0)
```

Levene's test for equal variances is significant, meaning we violate the assumption of homoscedasticity. Given that, the appropriate correlation test to use would be a non-parametric test such as the Spearman rank correlation or Kendall Tau correlation test. For demonstration purposes, I will still also conduct the Pearson correlation as well as the others.

Correlation Examples

The Pandas correlation method

To conduct the correlation test itself, we can use the built-in .corr() method which is apart of the pandas library. This method conducts the correlation test between the variables and excludes missing values for the variables being compared – this is called pairwise deletion. It can conduct the correlation test using a Pearson (the default method), Kendall Tau, and Spearman rank method. Official documentation is here.

```
df['carat'].corr(df['price'])
```

```
0.92
```

```
df['carat'].corr(df['price'], method= 'spearman')
```

```
0.96
```

You can see that the two different methods of calculating the level of correlation between the variables produces different results. Both methods show a strong level of correlation between the carat of the diamond and the price of the diamond. Unfortunately, the built-in pandas method does not provide a way to obtain the p-value for the correlation test. But fear not! There are other libraries with different methods that are available to use.

Pearson correlation method using scipy.stats.pearsonr()

To conduct the Pearson correlation test using scipy.stats, use the .pearsonr() method. Full documentation for this method can be found here. The output is not labelled, but it is returned in the order of (r-value, p-value).

```
stats.pearsonr(df['carat'], df['price'])
```

```
(0.92, 0.0)
```

The Pearson correlation indicates there is a statistically significant strong relationship between the price and carat of a diamond. Remember that this method of measuring correlation is not the measure to use since the data violated the assumption of homoscedasticity of variance.

Spearman rank correlation method using scipy.stats.spearmanr()

Now I will conduct a non-parametric measure of correlation which is better to analyze the relationship between the carat and price of a diamond. To do this using scipy.stats, use the .spearmanr() method. Full documentation on this method can be found here.

```
stats.spearmanr(df['carat'], df['price'])'])
```

```
(correlation= 0.96, pvalue= 0.0)
```

The Spearman rank correlation method indicates that the correlation is strong and significant between the size of the carat and the price of the diamond.

Kendall Tau correlation method using scipy.stats.kendalltau()

To conduct the Kendall Tau correlation measure using scipy.stats, us the .kendalltau() method. Full documentation on this method can be found here.

```
stats.kendalltau(df['carat'], df['price'])
```

```
(correlation= 0.83, pvalue= 0.0)
```

Using this method, the test indicates that there is a strong, significant correlation between the size of the carat and the price of the diamond.

Correlation using researchpy

The pandas built-in correlation methods are able to conduct pairwise correlation measures on multiple variables at a time and will return the results in a correlation matrix. However, this method does not produce p-values that are associated with each measure of correlation. The scipy.stats correlation measures are only able to conduct measures of correlation on pairs of variables at a time.

Now I will demonstrate correlation using researchpy which has the benefits of being able to run measures of correlation on pairs or multiple pairs of variables, calculates the p-value, and if running correlation on multiple pairs, you can decide if you want to use pairwise or casewise deletion methods. Full documentation can be found here. Currently, both the casewise and pairwise methods support Pearson, Spearman rank, and Kendall Tau measurements of correlation with Pearson being the default method.

CASEWISE

The casewise method returns all information in 3 dataframes. I will store each dataframe as its own object and display them individually for cleanliness of presentation.

```
corr_type, corr_matrix, corr_ps = rp.corr_case(df[['carat', 'price',
'depth']])


corr_type
```

Pearson correlation test using list-wise deletion
Total observations used = 53940

Now to see the actual correlation matrix itself.

```
corr_matrix
```

	carat	**price**	**depth**
carat	1	0.9216	0.0282
price	0.9216	1	-0.0106
depth	0.0282	-0.0106	1

Finally, the corresponding p-values.

```
corr_ps
```

	carat	**price**	**depth**
carat	0.0000	0.0000	0.0000
price	0.0000	0.0000	0.0134
depth	0.0000	0.0134	0.0000

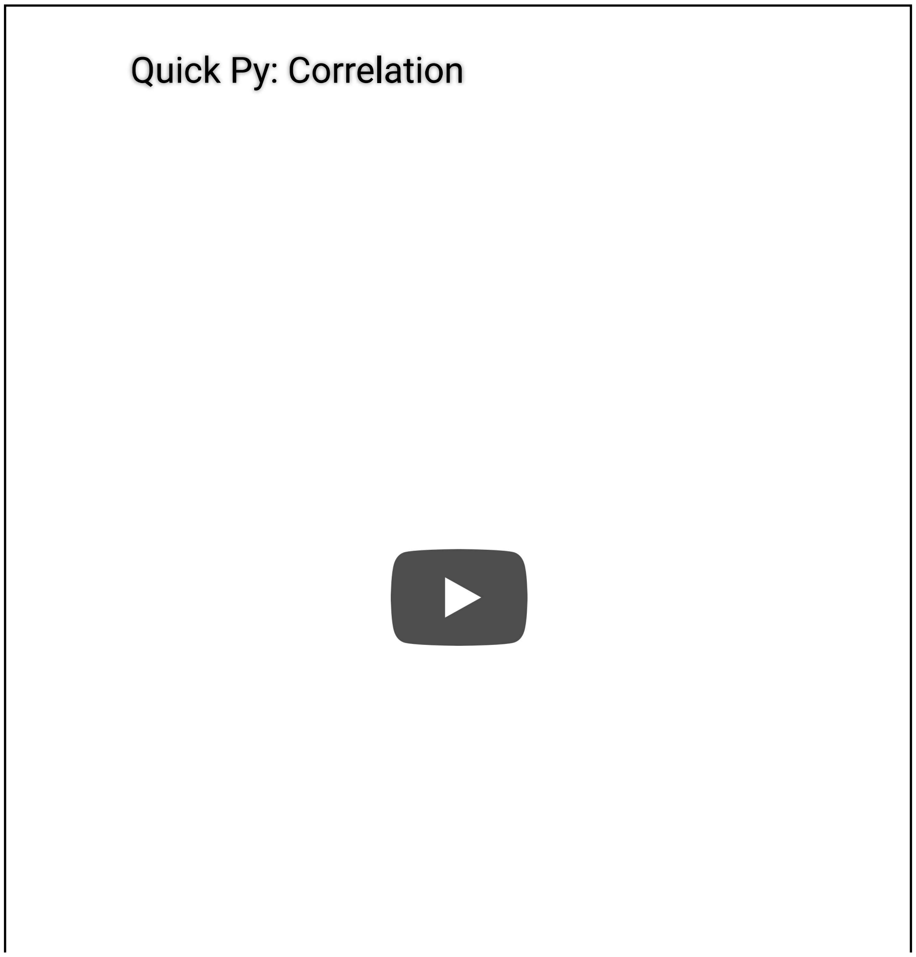
PAIRWISE

This method returns the information in a single dataframe and has a rather nice output.

```
rp.corr_pair(df[['carat', 'price', 'depth']])
```

	r value	p-value	N
carat & price	0.9216	0.0000	53940
carat & depth	0.0282	0.0000	53940
price & depth	-0.0106	0.0134	53940

How to Video



Share this:

Like this:

Loading...

3 thoughts on “Correlation”

Reshma Patel

July 13, 2018 at 5:16 pm

Excellent explanation and script. Its helped me lot during data pre-processing in my capstone project.....one minor change in script
import scipt.stats as stats
instead change it to
import scipy.stats as stats...
again superb video...

★ Loading...

July 13, 2018 at 8:11 pm

I’m glad this page was able to help you! Also, thanks for the editing catch; I made the changes.

Cheers,

Corey

★ Loading...

Reshma Patel

July 13, 2018 at 6:30 pm

Superb understanding of correlation...

★ Loading...

This site uses Akismet to reduce spam. Learn how your comment data is processed.